B.Baldin
09/05/00

Muon Front-End Parameters Setup

1. Introduction

This document describes the proposed procedures for downloading, setting and handling of muon front-end electronics parameters. It is based on discussions summarized on the muon electronics meeting held on August 15, 2000.

2. Description of the muon front-end electronics parameters

Muon front-end electronics includes three detector subsystems (PDT, SC and MDT) and Level 1 trigger subsystem. In the following discussion all muon subsystems considered identical unless otherwise noted. There are two types of the muon parameters:

- Hardware parameters
- Software parameters

Hardware parameters control the status of the hardware in the subsystem. These parameters may not change significantly between runs, but will be adjusted very often during commissioning period.

Software parameters exclusively describe the data processing algorithms in the particular subsystem. These parameters may significantly vary from run to run. Most of these parameters are used by the digital signal processors (DSP) in the front-end electronics.

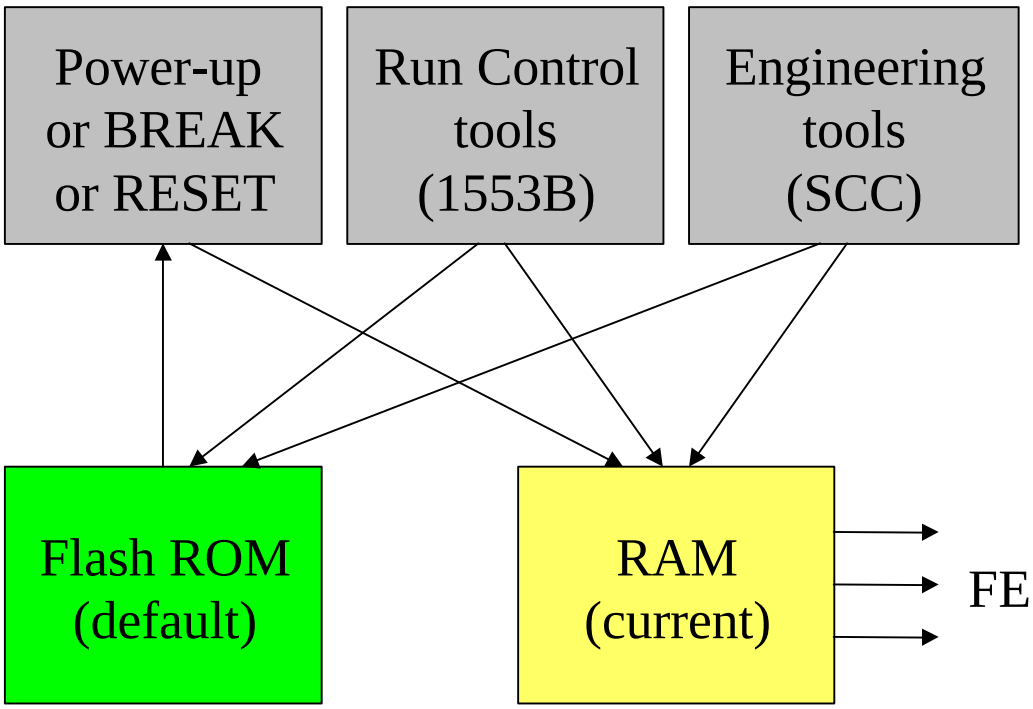

Fig. 1 Three ways to control muon parameters

All parameters (hardware and software) are remotely controlled via several user options. The latter include power-up reset, generation of the BREAK or manual subsystem reset, 1553 data bus interface and muon engineering tools (See Fig. 1). There are two storages for hardware parameters and three storages for software parameters. Default values of hardware and software parameters are stored in the permanent memory (FlashROM) for initial subsystem setup. These parameters are loaded into subsystem RAM during power up or after system reset (VME) or after BREAK character is received by the subsystem from the muon readout crate (MRC). After parameters are placed in the subsystem RAM, they can be changed by using 1553 bus interface (EPICS) or muon engineering tools (terminal port access). This set of parameters determines curent status of the subsystem. This allows remote control over the subsystem behavior. The software parameters are copied by the DSP into the third storage location upon reception of the INIT signal. DSP uses this only set of parameters for event data processing. Any time the hardware parameter is changed in the subsystem RAM, it immediately affects the corresponding hardware. Any change in the software parameter does not affect data processing until next INIT signal is issued by the muon readout crate. Corresponding sequence of changing muon parameters is shown in Fig. 2.

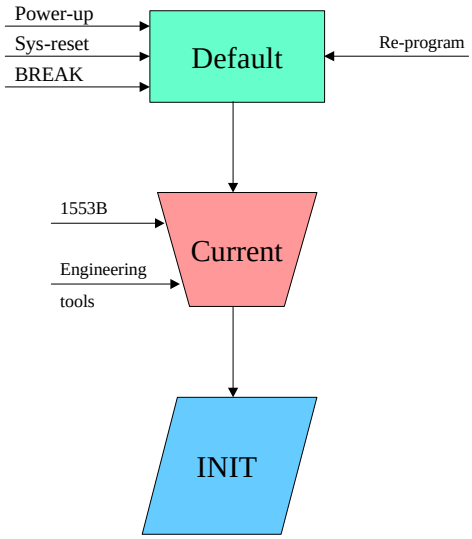

Fig. 2. Sequence of changing muon parameters

The default parameters can be changed by remote re-programming of the FlashROM using 1553 multi-block data transfers or engineering software tools. Page three (0..3) of the DSP FlashROM is used to store default parameters. The size of the page is 32 Kbytes. The structure of the parameters is system dependent. Note that INIT signal *does not change* any of the parameters.

3. BREAK signal implementation

The main purpose of the BREAK signal implementation is to guarantee synchronization between muon readout crate and front-end system. Every time muon readout code starts, it generates BREAK signal to all front-ends specified in the readout crate configuration file. The front-end system must perform equivalent of the power up reset and at the end of it respond to the muon readout crate with a single status character. The status character is defined as ASCII symbol “0” for failure and “1” for normal. If the front-end system detected an error condition and reported it to the readout crate, the corresponding error bit must be set in the 1553-readable status register. There are two ways to implement BREAK for generating system reset. The preferred way is to permanently connect INT pin of the Am85C30 SCC to the power up reset logic. The SCC must be programmed that way that it generates interrupts only on BREAK signal. Please note that the SCC will generate an interrupt for each transition of the BREAK. Right now we plan to send a short (< 10 μS) BREAK signal. The other possibility is to implement software control over the system reset. This is less desirable option since it may fail. In this case additional hardware reset function has to be implemented via 1553 bus to allow manual subsystem reset if it fails to respond to BREAK. The following are the default settings for the front-end SCC port connected to the readout crate:

- Asynchronous, 2.5 Mbit/s baud rate (40 MHz clock at RTxC pin)
- FM1 data encoding
- DPLL clock recovery for receiver
- 1x clock, 1 stop bit, no parity

These are default settings for all muon subsystems at the power up. These settings can be changed for a particular application to run locally (not during normal data taking), but restored back after the application has finished. The recommended sequence of commands is listed in Appendix 1 (Example provided by Sten Hansen).

Appendix 1. Programming sequence for Am85C30

Do this before loading all the registers with setup values.

```
AR = IO(B_Ctrl);        { Read RR0 }
AR = 1;
IO(B_Ctrl) = AR;        { Set pointer to 1 }
AR = IO(B_Ctrl);        { Read RR1 }
AR = 9;                 { Set Pointer to Reg 9 }
IO(B_Ctrl) = AR;
AR = 0xC0;              { Issue reset to both channels of SCC }
```

{**** Initialization of AM85C30 SCC ****}
{ Put SCC control register address and setup data into one 24 bit word:
0 x Ad Ad Dat Dat 0 0 }

{ Setup channel B as terminal port. 9600 baud, NRZ, clock source external 7.3728 MHz crystal }

```
.INIT SetUp_Vals:
    0x044400,       { Reg 4 : 16X Clock Mode, 1 Stop Bit, No Parity }
    0x03C000,       { Reg 3 : 8 bits/char }
    0x056000,       { Reg 5 : 8 bits/char }
    0x090000,       { Reg 9 : No IRQs }
    0x0A0000,       { Reg 10: NRZ protocol }
    0x0BD600,       { Reg 11: RxClk=BRG, TxClk=BRG,TRxC Out=off}
    0x0C1600,       { Reg 12: BRG time constant LSB }
    0x0D0000,       { Reg 13: BRG time constant MSB }
    0x0E0000,       { Reg 14: BRG Source = Xtal }
    0x0E0100,       { Reg 14: Enable BRG }
    0x03C100,       { Reg 3 : 8 bits/char, enable Rx}
    0x056800,       { Reg 5 : 8 bits/char, enable Tx}
```

{ Setup channel A as remote port. 2.5 Mbaud, FM1, clock source external 40 MHz oscillator, interrupt on receipt of break character }

```
    0x040400,       { Reg 4 : 1X Clock Mode, 1 Stop Bit, No Parity }
    0x03C000,       { Reg 3 : 8 bits/char }
    0x056000,       { Reg 5 : 8 bits/char }
    0x090000,       { Reg 9 : No IRQs for now }
    0x0A4000,       { Reg 10: FM 1 protocol }
    0x0B7700,       { Reg 11: RxClk=DPLL, TxClk=BRG,TRxC Out=DPLL}
    0x0C0200,       { Reg 12: BRG time constant LSB (PCLK/8) }
    0x0D0000,       { Reg 13: BRG time constant MSB }
    0x0EA000,       { Reg 14: DPLL Clk = RTxC }
    0x0D0000,       { Reg 13: BRG time constant MSB }
    0x0EA000,       { Reg 14: DPLL Clk = RTxC }
    0x0EC000,       { Reg 14: DPLL Mode = FM }
```

```
        0x0E2300,       { Reg 14: Start DPLL, BRG src=PClk,BRG on}
        0x090A00,       { Reg 9 : Global Enable Interrupts, No Vector }
        0x0F8000,       { Reg 15: Enable Break/Abort External Status Interrupt
}
        0x010100,       { Reg 1 : External Status IRQ on, Rx and Tx IRQs off }
              {There is a pending interrupt as soon as the interrupts are
              enabled. Both of the following two writes are needed to clear
              the pending interrupt.}
        0x001000,       { Reg 0 : Reset Ext/Status Interrupt }
        0x003800,       { Reg 0 : Clear Interrupt Under Service (IUS) Bit }
        0x03C100,       { Reg 3 : 8 bits/char, enable Rx}
        0x056800;       { Reg 5 : 8 bits/char, enable Tx}
```

Note: When writing to register zero, the step of setting the address pointer before sending data is skipped. If not used, the pin IntAck on the SCC must be pulled high.